

# UbiComp – Teil 8: Ethernet, OPC UA & MQTT

**Prof. Dr.-Ing. Dorothea Schwung**

# Lernziele Teil 8

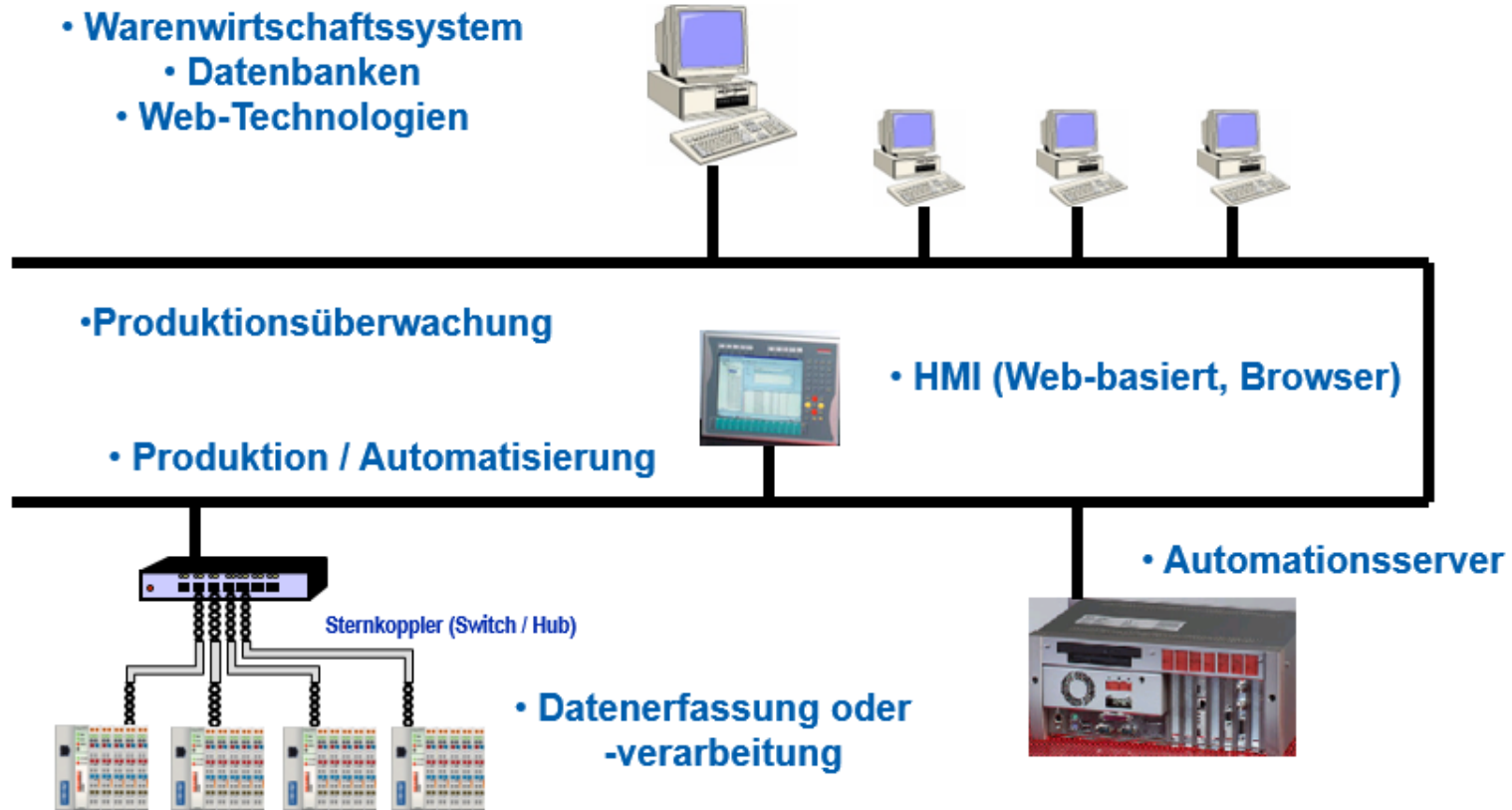
1. Sie sind mit der Arbeitsweise von Ethernet vertraut.
2. Sie verstehen die Unterschiede zwischen IP- und TCP-Protokollen.
3. Sie verstehen die Motivation für den Einsatz von OPC UA.
4. Sie besitzen ein Grundverständnis für die Struktur und Architektur von OPC UA.
5. Sie können die Arbeitsweise von MQTT erläutern.

# Ethernet-basierte Feldbusse – Eine Übersicht

**Modbus-IDA**



# Motivation – Vertikale Integration mit Ethernet



# Motivation – Vertikale Integration mit Ethernet

- **Vertikale Integration: Ethernet von der Leitebene bis zum Sensor/Aktor**
- **Ethernet mit Standardprotokollen (TCP/IP, UDP, Modbus TCP...)**
  - Protokolleigenschaften/Kollisionsproblematik
  - Stacklaufzeiten
  - Standard Applikation: Office Umgebung
- **Realtime Ethernet**
  - Siemens ProfiNet
  - B&R Powerlink (ABB)
  - Sercos III
  - Beckhoff EtherCAT

# Unterschied von Office zu Automation Netz

## ➤ Office

- Feste Installation
- Vorkonfektionierte Kabel
- Große Datenpakete (z.B. Dateien)
- Relativ lange Übertragungszeiten
- Nachrichten- bedarfsorientierter Datentransfer
- Keine hohen Umweltansprüche (Temperatur, Feuchte, EMV...)

## ➤ Automation

- Nicht immer feste Installation
- Evtl. Keine vorkonfektionierten Kabel verwendbar
- Mehr zyklische Datenübertragung, aber auch azyklisch
- Hohe Umweltansprüche (Temperatur, Feuchte, Erschütterungen, EMV)

# Unterschied von Office zu Automation Netz

## ➤ Antriebstechnik

- Zykluszeit bis auf  $125\mu\text{s}$  (→ Echtzeitanforderung)
- Hoher Determinismus
- Synchronität
- Viele Teilnehmer (Anlage mit bis 100 Achsen/PC)
- Vielfach großer Abstand zwischen den Antrieben (bis 100m)
- Neben den zyklischen Daten viele azyklische Parametrierdaten
- Flexible Topologie
- Besonders wichtig: hohe Diagnoseabdeckung!

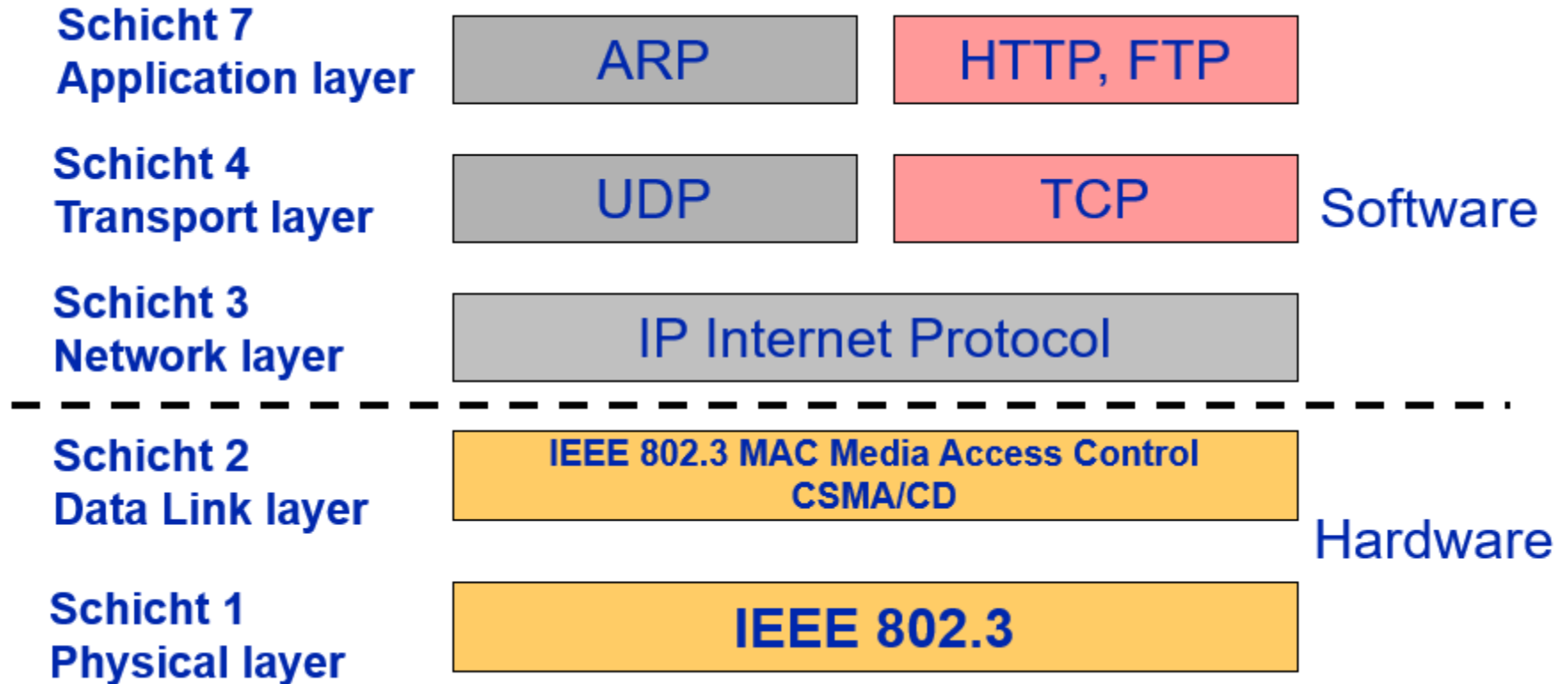
# Netzkategorien

<b>LAN</b> (Local Area Network)	<b>MAN</b> (Metropolitan Area Network)	<b>WAN</b> (Wide Area Network)
Begrenzung auf ein Firmen- oder Campusgelände mit ca. 500m Ausdehnung	Regionale Ausdehnung ca. 100 km	Unbegrenzte Ausdehnung

	<b>Private Netze</b>			<b>Öffentliche Netze</b>	
	<b>LAN</b>		<b>MAN</b>	<b>WAN</b>	
<b>Geschwindigkeit</b>	10 – 100 Mbit/s	1 Gbit/s	155 – 622 Mbit/s	Bis 64 kbit/s	Bis 2,5 Gbit/s
<b>Anwendungen</b>	Daten, Bilder	Daten, Multimedia-Anwendungen	Daten, Sprache	Daten, Bilder	Daten, Sprache, Video
<b>Technologie</b>	Ethernet, Fast Ethernet, Token Ring, HSTR	Gigabit-Ethernet, FDDI, ATM	ATM	Analog, ISDN	ATM



# Ethernet im ISO-OSI Schichtenmodell



# Übertragungsraten

	Ethernet	Fast Ethernet	Gigabit Ethernet
Übertragungsrate	10 Mbit/s	100 Mbit/s	1000 Mbit/s
Bitzeit	0.1 $\mu$ s	0.01 $\mu$ s	0.001 $\mu$ s
Medium	10Base-T u.a.	100Base-T u.a	1000Base-T u.a.
Encoding	Manchester	4B/5B	4D-PAM5 o. 8B/10B
Stecker	RJ-45	RJ-45	RJ-45
Topologie	Bus, Stern	Stern	Stern
Standard	IEEE802.3	IEEE802.3.u	IEEE802.3.z
Zugriffsverfahren	CSMA/CD	CSMA/CD	CSMA/CD
Min. Paketgröße	64 Byte	64 Byte	512 Byte

Dr. – Ing. Guido Beckmann, Lenze

# Telegrammaufbau

## Telegramm mit variabler Datenfeldlänge

PRE	DA	SA	Type	Datenfeld (min. 46 byte, max. 1500 byte)	Pad	FCS
8	6	6	2		0-46	4

- PRE** : Vorspann (Preamble)  
**DA** : Empfängeradresse (Destination address)  
**SA** : Absenderadresse (Source address)  
**Type** : Typnummer oder Länge des Telegramms (z.B. 0x0800 = IP)  
**Pad** : Ergänzungszeichen (padding bytes)  
**FCS** : Prüfbyte (Frame Check Sequence)

# Internetprotokollfamilie – TCP/IP-Referenzmodell

OSI-Schicht	TCP/IP-Schicht	Beispiel
Anwendungen (7)	Anwendungen	HTTP, UDS, FTP, SMTP, POP, Telnet, DHCP, OPC UA  TLS, SOCKS
Darstellung (6)		
Sitzung (5)		
Transport (4)	Transport	TCP, UDP, SCTP
Vermittlung (3)	Internet	IP (IPv4, IPv6), ICMP (über IP)
Sicherung (2)	Netzzugang	Ethernet, Token Bus, Token Ring, FDDI
Bitübertragung (1)		



Quelle: Wikipedia

# IP (Internet Protokoll)

- **Datenpaketversendung: Übermittlung von Daten vom Sender zum Empfänger**
- **Management von Adressen**
  - Momentan 4 stellig z.B. 172.16.17.4 (IPv4)
  - In Zukunft: 6-stellig! (IPv6)
- **Segmentierung von zu großen Telegrammen**
- **Netzwerkkontrollfunktionen**
  - Internet Control Message Protocol (ICMP bzw. ICMPv6)

Quelle: Wikipedia

ICMP (Internet Control Message Protocol)					
<b>Familie:</b>	Internetprotokollfamilie				
<b>Einsatzgebiet:</b>	Obligatorischer Zusatz zum Internet Protocol, Fehlermeldungen, Diagnose				
ICMP im TCP/IP-Protokollstapel					
Internet	ICMP				
	IPv4				
Netzzugang	Ethernet	Token Bus	Token Ring	FDDI	...
<b>Standards:</b>	RFC 792 <span>↗</span> (1981)				

IP (Internet Protocol)					
Familie:		Internetprotokollfamilie			
Einsatzgebiet:		Datenpaketversendung sowohl lokal als auch weltweit über verschiedene Netzwerke			
IP im TCP/IP-Protokollstapel:					
Anwendung	HTTP	IMAP	SMTP	DNS	...
Transport	TCP			UDP	
Internet	IP (IPv4, IPv6)				
Netzzugang	Ethernet	Token Bus	Token Ring	FDDI	...
Standards:		RFC 8200  (IPv6, 2017) RFC 791  (IPv4, 1981)			

Quelle: Wikipedia

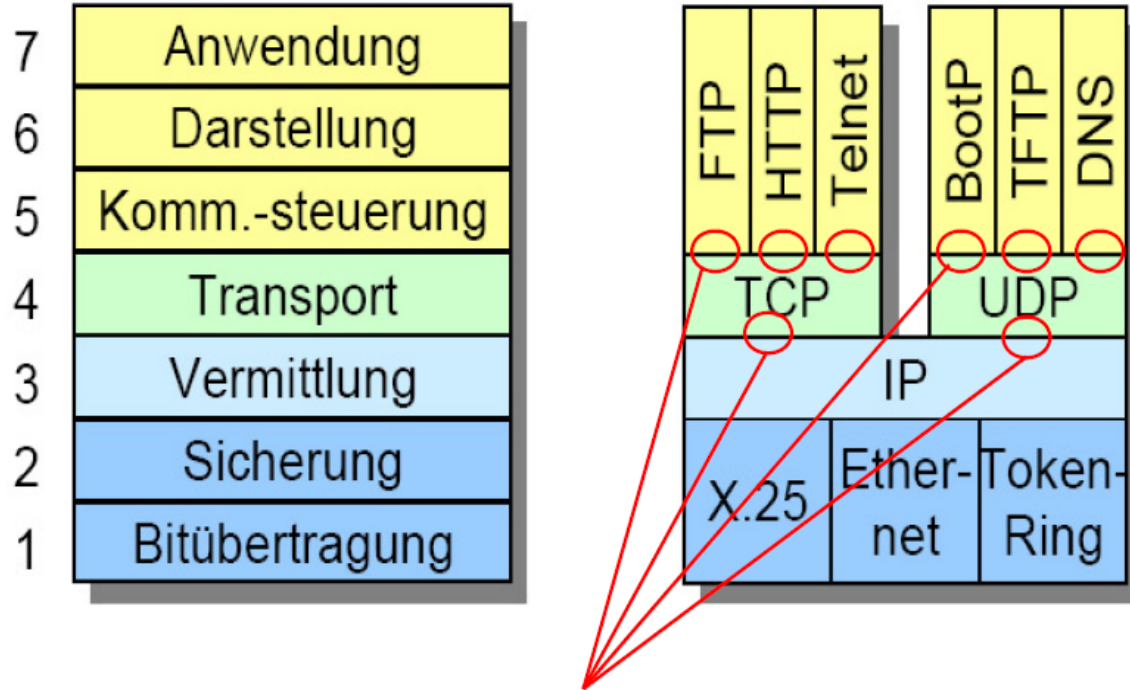
# TCP (Transmission Control Protokoll)

- **Zuverlässigkeit der Übertragung von Datenströmen**
  - Telegramme werden immer beantwortet
- **Verbindungsorientiertes Protokoll**
- **Verbindungsüberwachung**
- **Zwischenpufferung**
- **Ende-zu-Ende Verbindung in Vollduplex**
- **Portnummer (z.B. Port 80 = HTTP)**

TCP (Transmission Control Protocol)	
<b>Familie:</b>	Internetprotokollfamilie
<b>Einsatzgebiet:</b>	Zuverlässiger bidirektionaler Datentransport
TCP im TCP/IP-Protokollstapel:	
Anwendung	HTTP SMTP ...
Transport	TCP
Internet	IP (IPv4, IPv6)
Netzzugang	Ethernet Token Bus Token Ring FDDI ...
<b>Standards:</b>	RFC 793 (1981) RFC 7323 (2014)

Quelle: Wikipedia

# TCP-Protokoll - Ports



**Ports = Dienstzugangspunkte zwischen den Schichten**

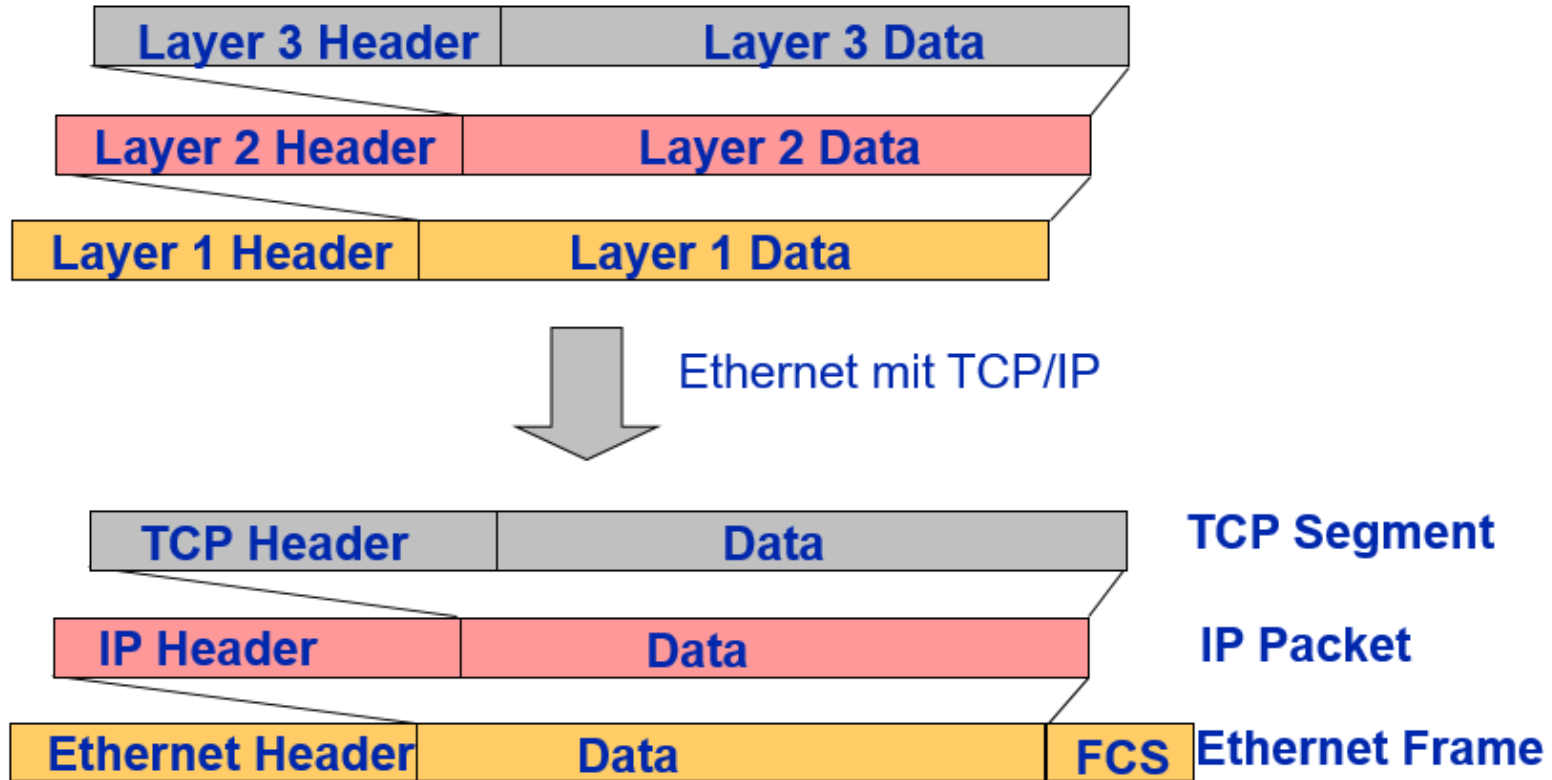
# TCP-Protokoll - Ports

## Beispiele für fest definierte Ports für TCP:

Port-Nummer	Dienst
23	Telnet
21	FTP
25	Simple Mail Transfer Protocol (SMTP)
53	Domain Name System (DNS)
80	HTTP-Webserver
502	Modbus



# Telegrammaufbau



# Möglichkeiten zur Echtzeitfähigkeit

## ➤ Zeitscheibenverfahren

- Polling: Ethernet Powerlink
- Spezielle Hardware: Siemens PROFINet V3

## ➤ Uhrensynchronisation (IEEE 1588)

- Uhren werden im Netzwerk synchronisiert
- Jedes Telegramm wird auf die globale Zeitbasis zurückgeführt

## ➤ Priorisierung von Nachrichten

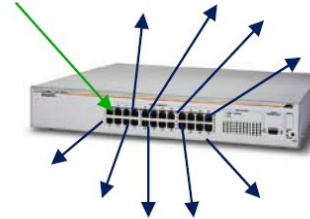
- IEEE 802.3 p/Q Standard
- Jedes Ethernet Telegramm kann damit priorisiert werden
- Siemens PROFINet V2, CIP-Sync

# Komponenten

## Komponenten zur Verbindung mehrerer Computer bzw. Netze

- **HUB:** Schicht 1
- **Switch/Bridge:** Schicht 2
- **Router:** Schicht 3
- **Gateway:** kann auf allen Schichten implementiert werden

• HUB:



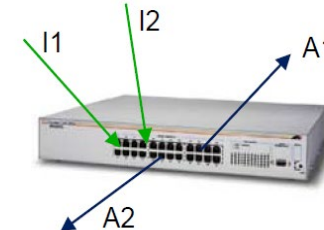
Eingehende Information



Ausgehende Information



• Switch:

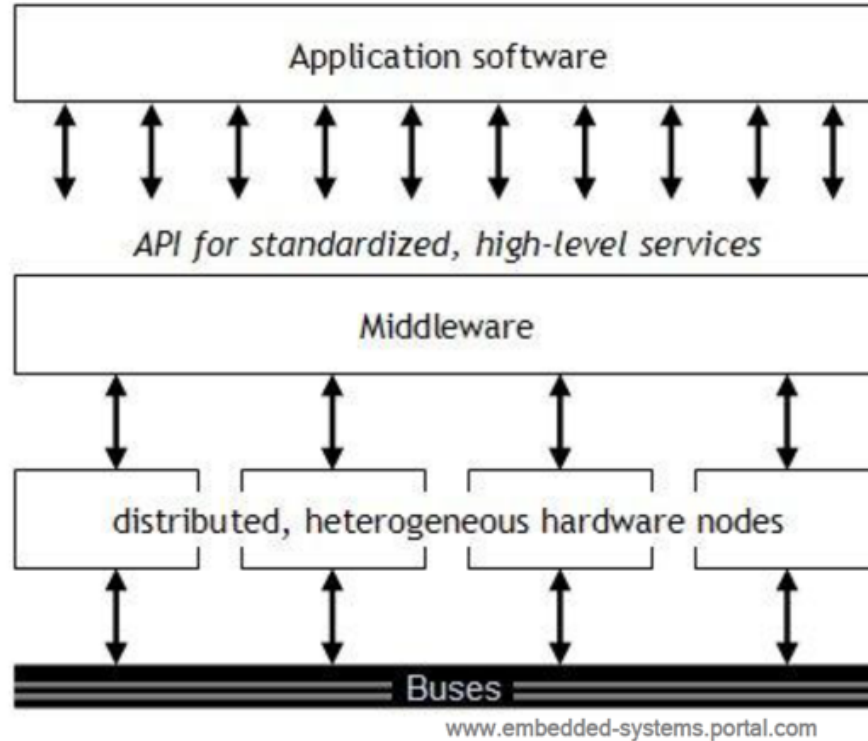


# Middleware für Automatisierungssysteme

- Middleware für Automatisierungstechnik [VDI/VDE 2657-1]: „**Softwareschicht** zur **Verbindung** von **Komponenten** einer automatisierungstechnischen Anwendung, beispielsweise Anlagen, Maschinen, Steuerungen, Leitsysteme oder Algorithmen.“

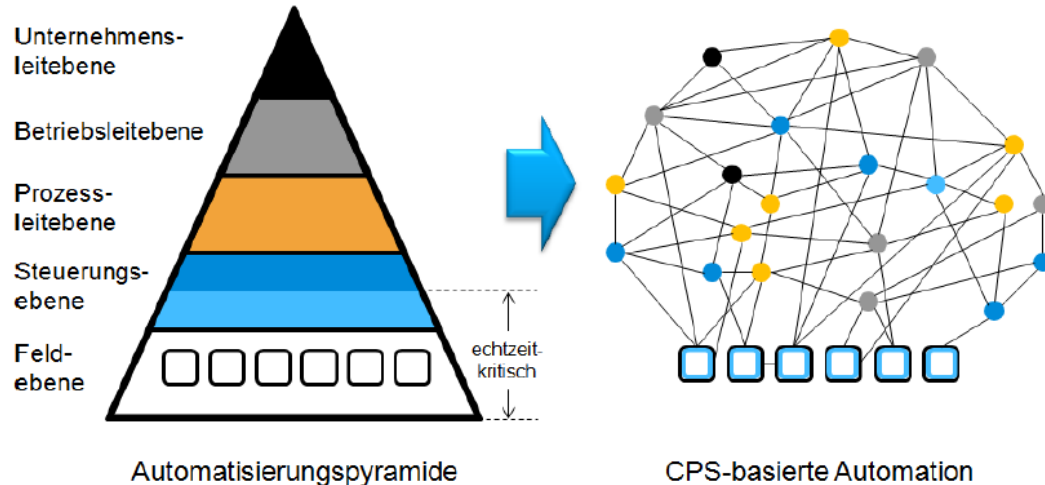
# Umgebung von Middleware

- Schnittstelle zwischen Softwaresystemen
- Keine direkte Nutzerinteraktion
- Skalierbarkeit der Hardware



# Ausgangssituation

- Vielzahl an Komponenten und Applikationen in Automatisierungssystemen
- Höhere Vernetzung auf Grund von
  - Steigender Komplexität
  - Zunehmende Leistungsfähigkeit einzelner Systemelemente
  - Starke Verbreitung des Internets
- Aufbruch der Hierarchie der Automatisierungspyramide



# Ausgaben und Funktionen

## ➤ **Applikationssicht:**

- Teilnehmerübergreifende, einfache und übersichtliche Anwendungslogik
- Programmierung ausschließlich über Middleware
- => Unabhängigkeit von konkreter Implementierung

## ➤ **Hardware-sicht:**

- Einfache Interfaces (enden an Middleware-Schnittstelle)
- Unabhängigkeit von Applikation

# Vorteile

## ➤ **Abstraktion**

- Einheitliche Abbildung gleichartiger Funktionen

## ➤ **Eindeutige Schnittstellen**

- Befördern arbeitsteilige Prozesse

## ➤ **Getrennte Entwicklung und Wiederverwertung** von Anwendungen und Schnittstellen

## ➤ **Vereinfachung und Reduktion** der zu entwickelnden Schnittstellen

## ➤ **Geringer anwendungsspezifischer Anteil**

- Übersichtlichkeit
- Flexibilität
- Erweiterbarkeit
- Wartbarkeit
- Zuverlässigkeit



# Abbildung des Daten- und Informationsraums

## ➤ **Definition der auszutauschenden Daten**

- Syntax (transparent für Nutzer)
- Semantik

## ➤ **Fest definierte Modelle**

- Einfacher Einstieg der Anwendungsentwicklung
- Semantische Erosion („Missbrauch“ von Feldern)

## ➤ **Generische Modelle**

- Erweiterbarkeit des Informationsmodells
- Abstraktion
- Modellierung durch Betreiber

# Standardisierung - Beispiele

- **IEC 62424 - CAEX**
  - Informationsaustausch zwischen Verfahrens- und Automatisierungstechnik
  - z.B. durch SignalInterface, ProcessConnectionInterface etc.
- **ISO 15926 - Lifecycle Information Exchange**
  - Lebenszyklus einer verfahrenstechnischen Anlage
- **IEC/DIN EN 61131-3 - SPS**
- **IEC 62264 (ISA-95)**
  - Modellierung von Produktionsprozessen
- **Beispiel für industrielle Middleware: OPC UA**

# Was bedeutet Industrie 4.0?

Der Begriff **Industrie 4.0** steht für die vierte industrielle Revolution, einer **neuen Stufe der Organisation und Steuerung der gesamten Wertschöpfungskette** über den Lebenszyklus von Produkten.



**Paradigmenwechsel kennzeichnen die Produktion in Industrie 4.0:**

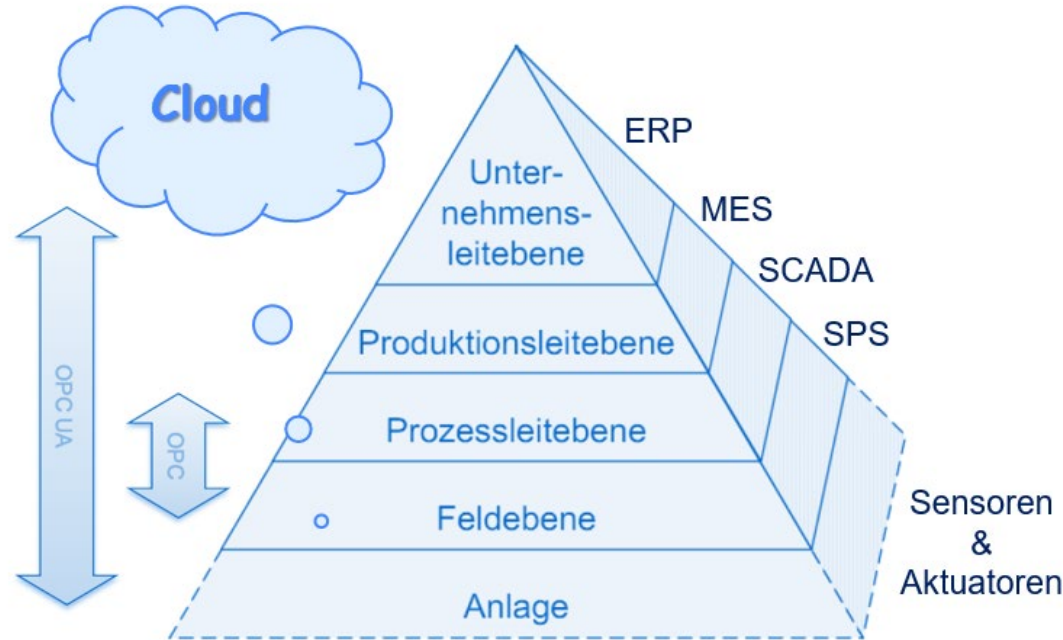
- Zentrale Steuerung – starr & komplex
- Deterministische Entscheidungen
- Etablierte Wertschöpfungsketten
- Vorgeplant betriebene Produktionssysteme
- Erweiterung durch Upscaling
- Werkstücke/Produkte sind passive Objekte der Bearbeitung
- Starre Anwesenheit der Mitarbeiter
- Dezentrale Selbstorganisation durch Ad-hoc Vernetzung
- Kontextabhängige Entscheidungen auf Basis von Echtzeitsimulation
- Virtuelle Ad-hoc Organisationen & Wertschöpfungsnetze
- Autonome, sich selbst organisierende Produktionseinheiten
- Erweiterung durch Upnumbering (Modularisierung)
- Intelligente Werkstücke/Produkte unterstützen aktiv den Produktionsprozess
- Flexibler Einsatz der Mitarbeiter

➤ **Auflösung der klassischen Branchen- & Domänengrenzen hin zu Interoperabilität**

# Die Automatisierungspyramide in der Industrie 4.0

**Horizontale Integration:** Regler-zu-Regler bzw. Maschine-zu-Maschine (M2M)

**Vertikale Integration:** von Sensoren/Aktuatoren und Reglern auf Feldebene zu IT-Systemen oder einer Cloud und vice versa

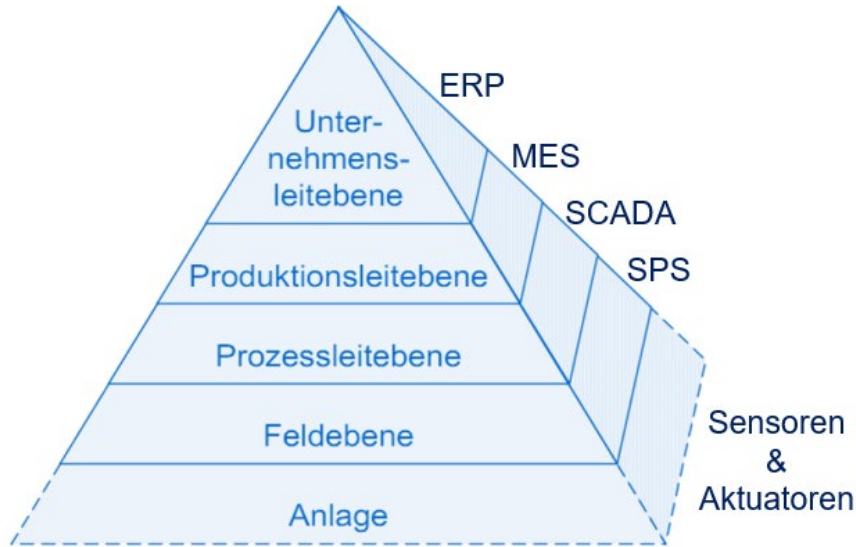


# Merkmale einer Smart Factory

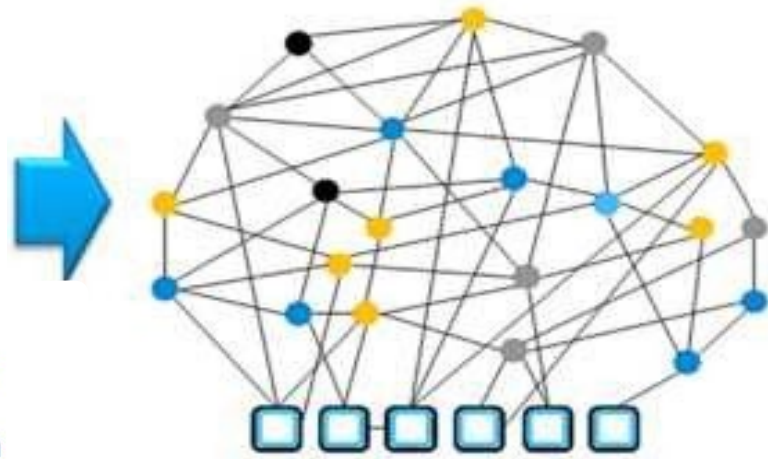
## Smart Factory:

- Wandlungsfähigkeit & Flexibilität → Produktion in Losgröße 1
- Ressourceneffizienz → Nachhaltigkeit & Umweltfreundlichkeit
- Durchgängigkeit vom Design bis zum Produkt
- Automatisierung verbessern:
  - Verfahren zur Selbstoptimierung
  - Selbstkonfiguration
  - Selbstdiagnose
  - Kognition
- Integration von Kunden & Geschäftspartnern in die Geschäfts- & Wertschöpfungskette

# Merkmale einer Smart Factory



Automatisierungspyramide



CPS-basierte Automation

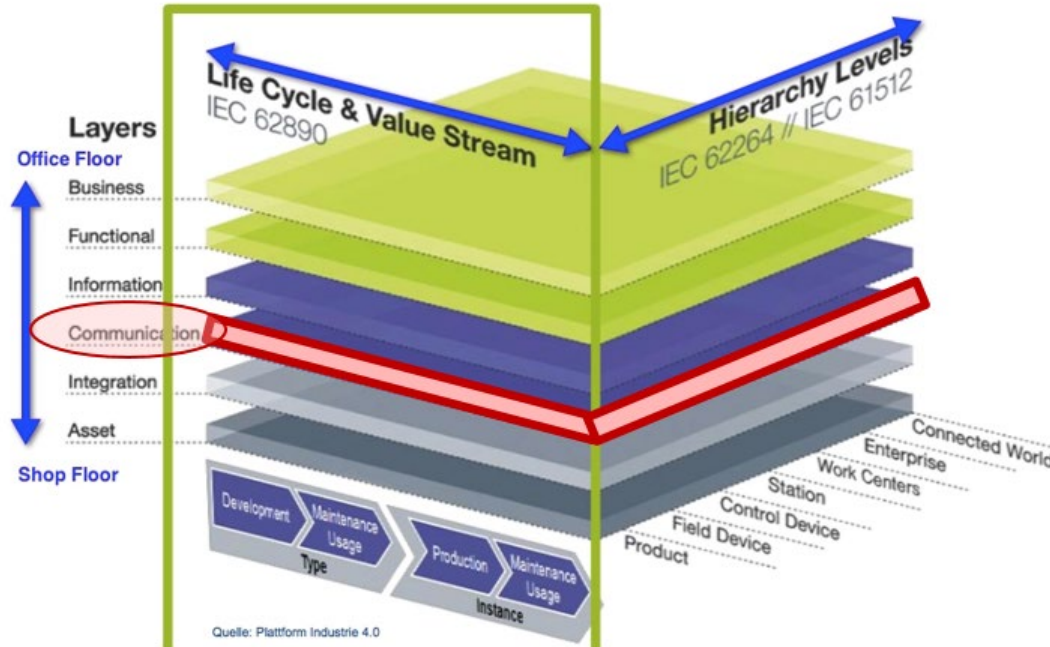
# Herausforderungen einer Smart Factory

## Herausforderungen:

- Im Bestand viele Komponenten & Anwendungen unterschiedlichster Hersteller
- Unterschiedliche Standards, z.B. bei Feldbussen
- Zuverlässige & sichere Kommunikation (in Echtzeit) zur Realisierung einer Smart Factory nach Industrie 4.0 Merkmalen

➤ **Lösung: Weitgehende Standardisierung durch Referenzarchitekturen, Kommunikationsstandards und Datenmodelle**

# RAMI 4.0 – Referenzarchitekturmodell für Industrie 4.0



- **Lösung:**  
**OPC UA als domänenübergreifender & herstellerunabhängiger Kommunikationsstandard**



# OPC UA – Ursprung & Generelles

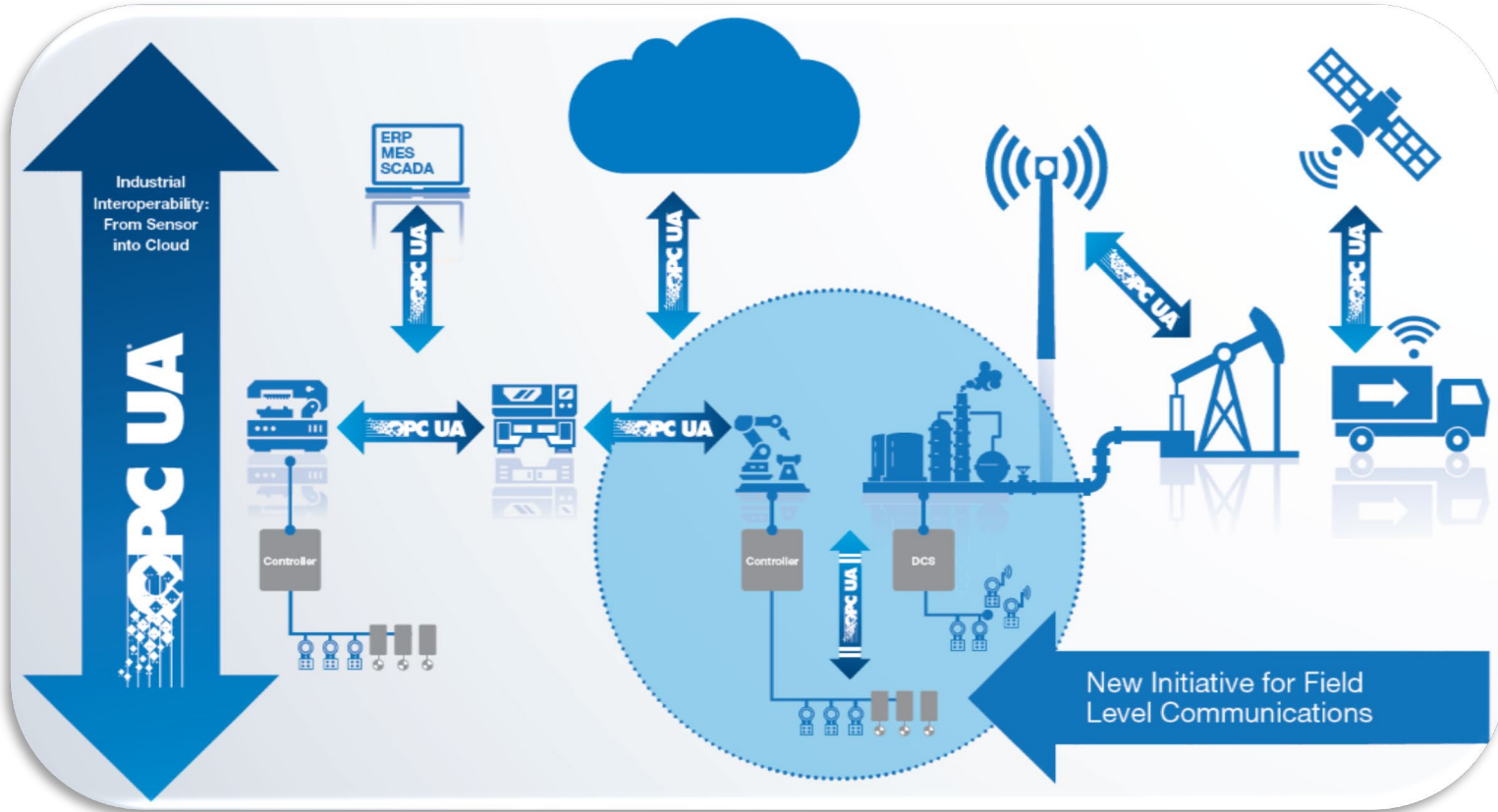
- **OPC UA** (Open Platform Communications Unified Architecture)

- Fügt sich ein in das Referenzarchitekturmodell für Industrie 4.0 (RAMI4.0)
- Arbeitet mit Service-orientierten Architekturen (SoA), die s.g. Dienste über eine Server-Client-Struktur zur Verfügung stellt
- IP-fähiges Netzwerk ist Grundvoraussetzung
- Nicht nur Datentransport, sondern auch maschinenlesbare semantische Beschreibung durch Modelle

- **OPC Foundation**

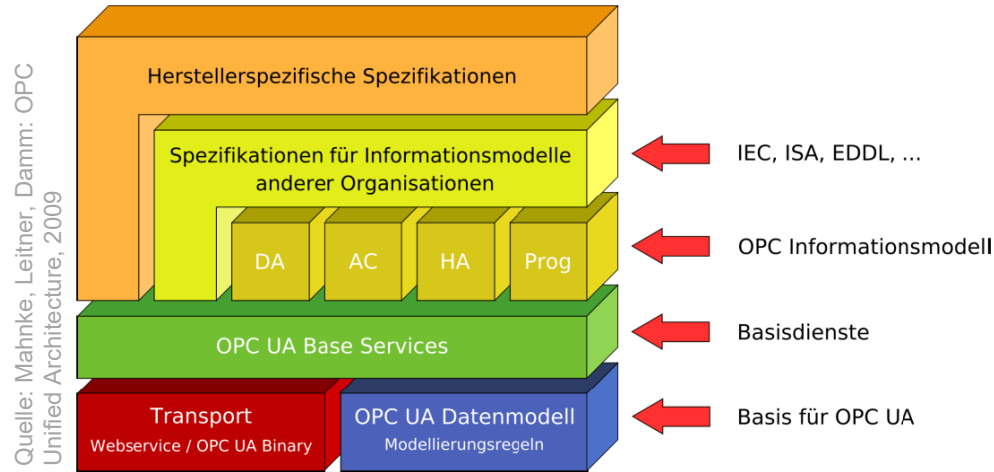
- Derzeit ca. 800 Mitglieder, darunter ABB, Siemens, Hitachi, Honeywell, Microsoft, Yokogawa...
- Erarbeitet offenen Standard spezifiziert in der IEC 62541, der alle Anforderungen an die Industrie-4.0-Kommunikation erfüllt, erste Version 2006
- Enge Kooperation mit anderen Organisationen, deren Informationsmodelle über OPC UA transportiert werden sollen (Companion Spezifikationen)

# Die Dimensionen von OPC UA



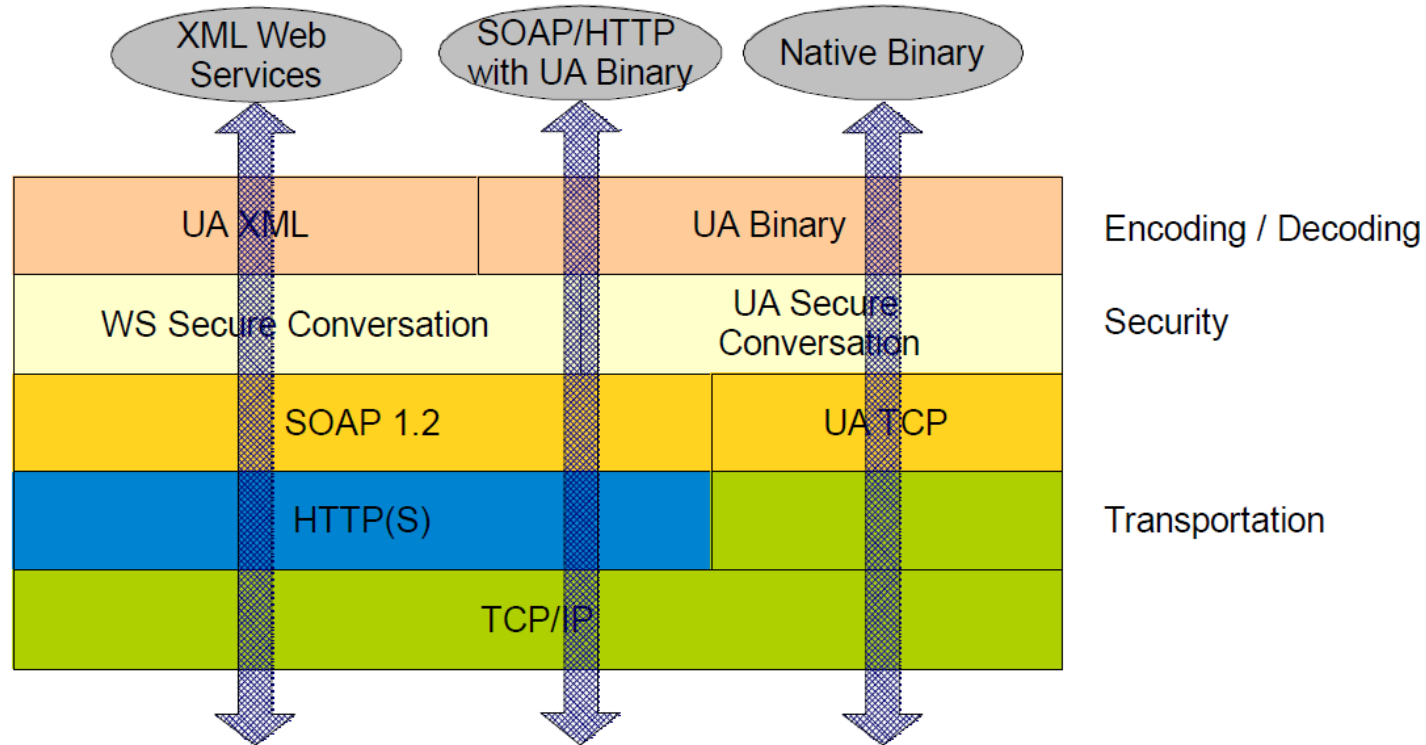
# OPC UA – Architektur

Die OPC-UA-Struktur arbeitet im Wesentlichen als SoA und besteht aus mehreren Schichten.



- Eigenschaften:**
- Vereinheitlichung aller Spezifikationen
  - spezifikationsübergreifendes Datenmodell
  - Einheitlicher Namensraum
  - Hoch performantes Binärprotokoll
  - Echte Security
  - Erweiterbare Basis-Services
  - Schlanke ANSI-C Implementierung für Embedded-Systeme

# Transport – Technology Mapping



Quelle: Urbas, TU Dresden, 2013

# Transport – Eigenschaften

- **XML Web Services**

- Kompatibilität durch Verwendung der W3C-Standards
- Verbindung über HTTP(S) Port 80(443)
- Relativ großer Overhead, geringe Performance

- **Native Binary**

- Genau spezifiziert, daher hohe Interoperabilität mit Geräten, die das Protokoll beherrschen
- Hohe Performance, geringer Ressourcenverbrauch
- Verbindung über TCP Port 4840 bzw. über Tunnel

- **SOAP/HTTP mit UA Binary**

- Kompromisslösungen zwischen Kompatibilität und Performance

- **OPC UA over TSN** (Time-Sensitive-Networking)

- Echtzeitfähigkeit mit Publish/Subscribe-Modell (PubSub) für many-to-many Kommunikation

# Sicherheitsarchitektur



Quelle: [www.opcfoundation.org](http://www.opcfoundation.org)

# Sicherheitseigenschaften

- **UA Security**

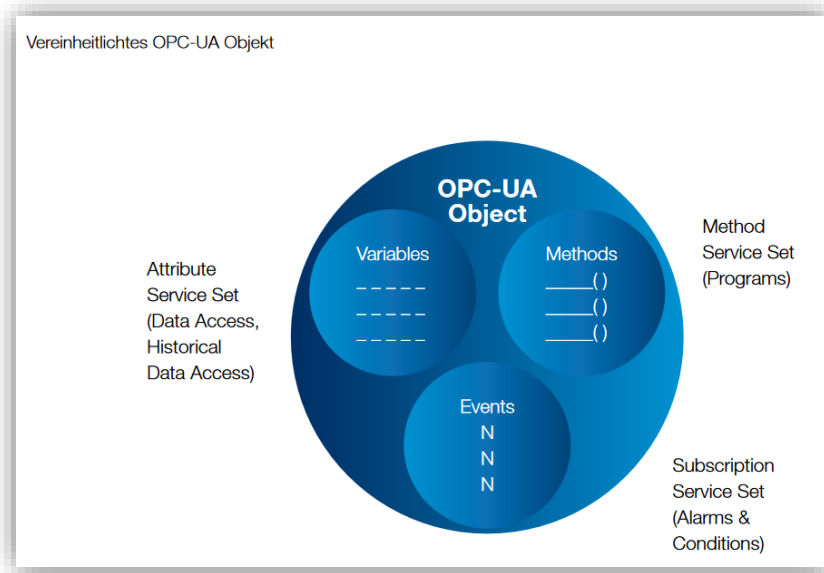
- WS Secure Conversation oder UA Secure Conversation als binäres Äquivalent
- Authentifizierung, Autorisierung, Vertraulichkeit, Datenintegrität, Prüfbarkeit und Verfügbarkeit auf Basis der Web Service Security Spezifikationen
- Security auf Transport Level (**Verschlüsselung**), Application Level (**Zertifikate**) und User Level (**Benutzerauthentifizierung**)

- **Robustheit**

- Verbindungsüberwachung in beide Richtungen (Heartbeat, Acknowledgement)
- Datenpufferung und Redundanz, sodass Verbindungsunterbrechungen nicht mehr zu Datenverlust führen

# Datenmodell - Objektmodell

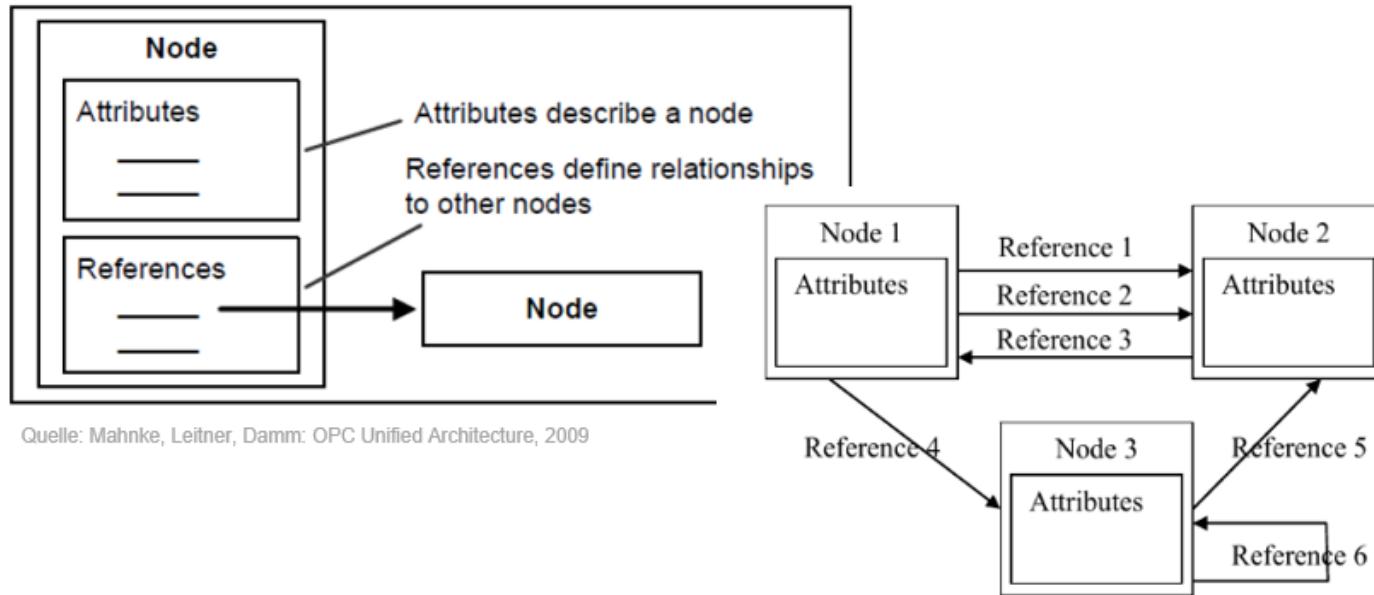
- Sämtliche Elemente im OPC UA Adressraum sind Objekte.
- Objekte werden definiert durch Variablen, Methoden und Events.
- Objekte und deren Komponenten werden als ein Satz von Knoten repräsentiert.





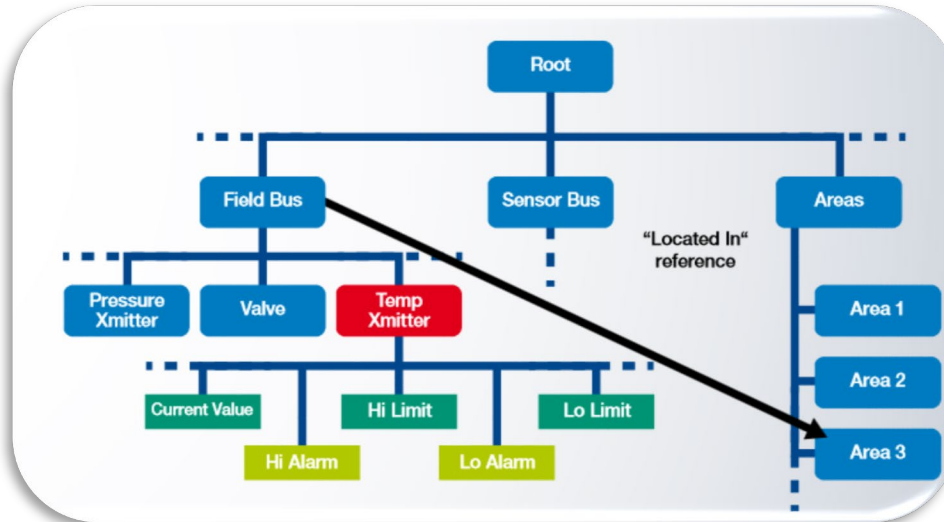
# Datenmodell - Knotenmodell

- Jeder Knoten gehört einer Knotenklasse an, die jeweils ein bestimmtes Element des Objektmodells darstellt.
- Knoten werden durch Attribute beschrieben und durch Referenzen verknüpft.



# Datenmodell - Adressraum

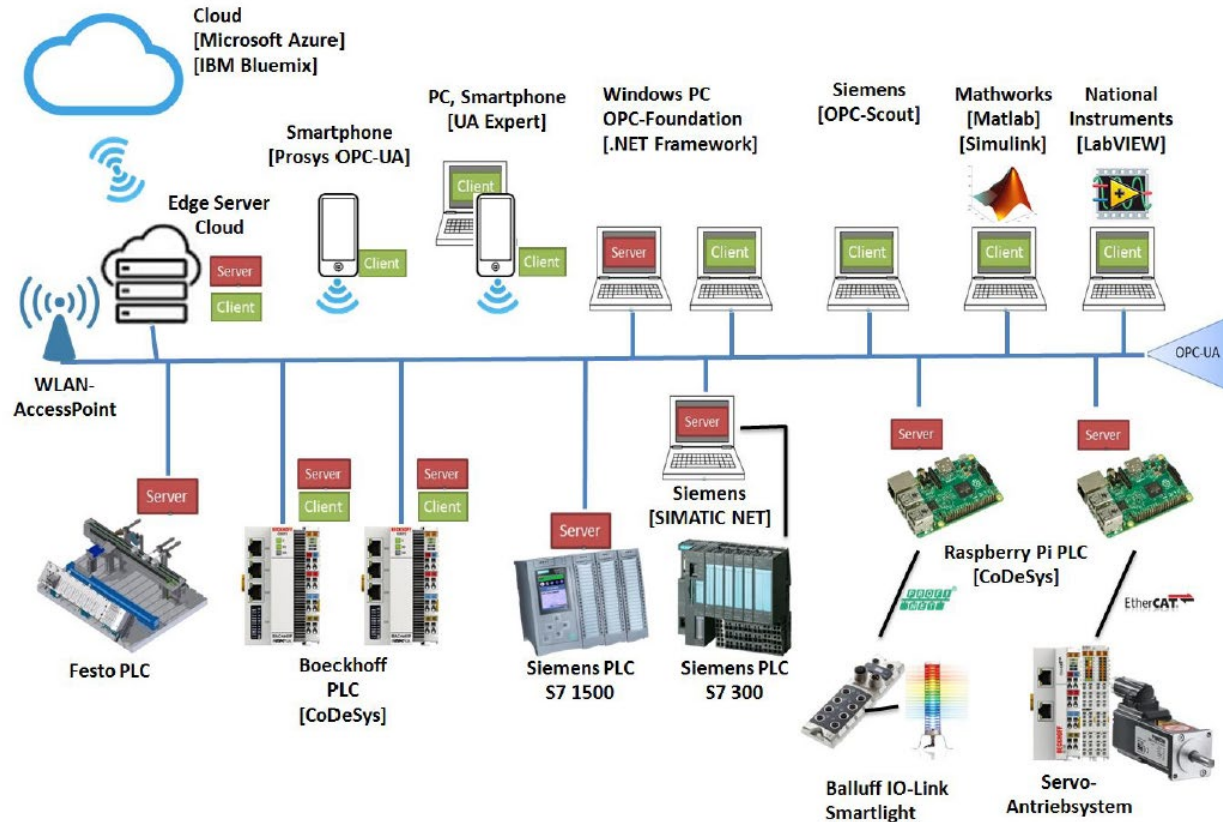
- Sämtliche Objekte werden durch Knoten im Adressraum dargestellt.
- Der Server kann Teile des Adressraumes als eine View definieren und bestimmten Clients verschiedene Views anbieten.
- Der Adressraum ermöglicht einen einheitlichen Zugriff auf Daten (DA), Historische Daten (HA), Alarmer und Zustände (AC) sowie auf Funktionen (Prog).
- OPC UA ermöglicht es, Knoten innerhalb des Adressraumes sowie innerhalb eines anderen Adressraumes zu referenzieren.



# Base Services – Die Welt der Dienste

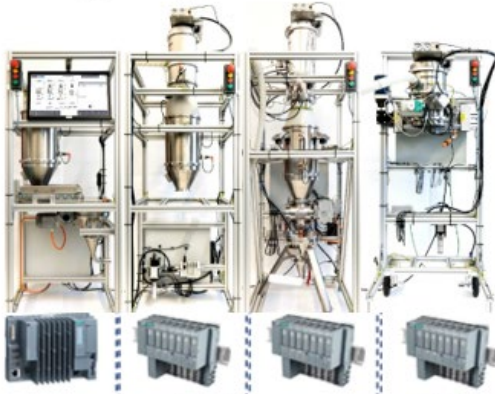
- OPC UA Service Definitionen sind abstrakte Methodenbeschreibungen und protokollunabhängig.
- Ein Service wird definiert durch seine Request- und Response-Messages.
- 34 definierte Services sind in 9 Service Sets organisiert.
- Sämtliche verfügbaren Services eines Servers sind im Server-Profil definiert.

# Anwendungsbeispiele – Eine Übersicht

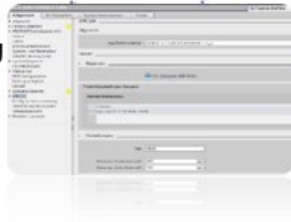


# Ein ausgewähltes Beispiel

Schüttgutanlage



Siemens CPU  
als  
Server



OPC UA

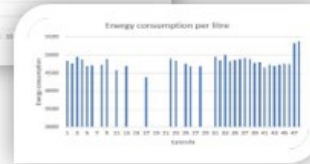
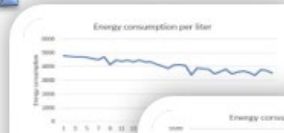
Mathworks  
[Matlab]  
[Simulink]



Umsetzung

```
serverList = opcaserverinfo('IP-Adresse und Port der OPC eintragen');  
hInfo = findDescription(serverList, 'Applikationsname eintragen');  
uClient = opca(hInfo);  
connect(uClient);  
browseNameSpace(uClient);  
Sample = opcasnode(6, 'Data', 'Sample');  
[val_Sample, ts_Sample, qual_Sample] = readValue(uClient, Sample);  
writeValue(uClient, Sample, 1); // Schreibe 1 in Variable Sample  
x = sprintf('%d %d', val, val);
```

Darstellung,  
Analyse  
&  
Weiterverarbeitung



# MQTT – Was ist das?

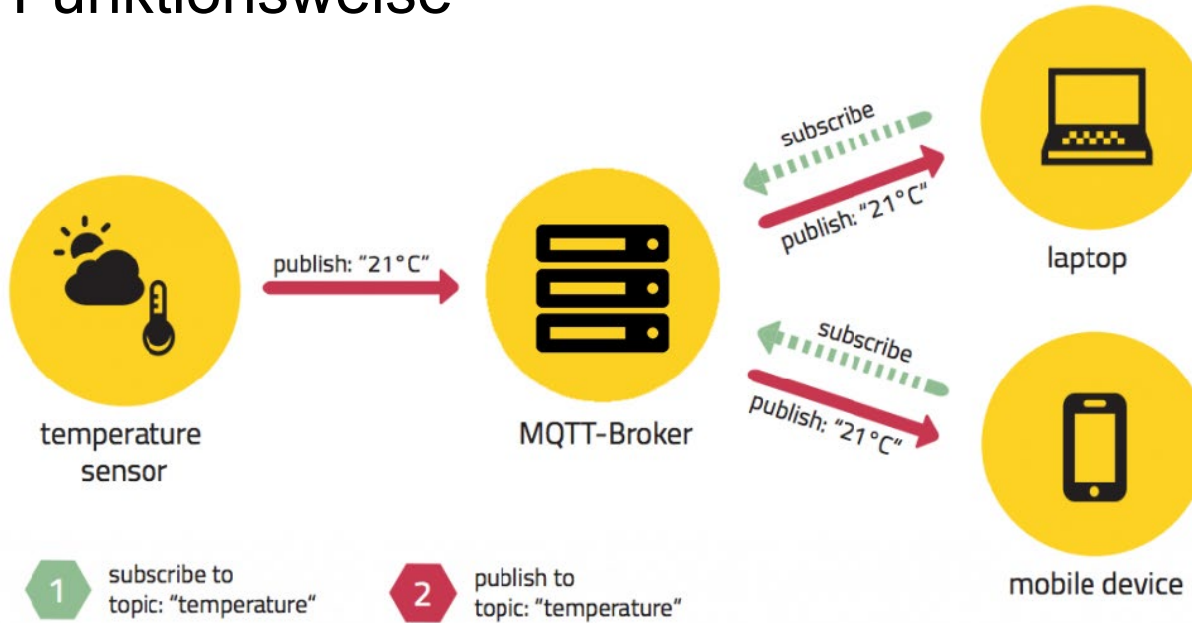
## MQTT (→ Message Queue Telemetry Transport Protocol)

- ist ein ursprünglich von IBM entwickeltes **leichtgewichtiges Kommunikationsprotokoll** für **M2M-Kommunikation im Internet of Things auf Anwendungsebene**.
- wurde dafür entwickelt, an Orten **mit limitierten Infrastrukturnetzen** und **energiesparenden Geräten mit einer begrenzten Leistung** eine zuverlässige Nachrichtenvermittlung zu ermöglichen (**Telemetrie-Daten-Übermittlung**).
- wird mittlerweile von der *Organization of the Advancement of Structured Information Standards* (OASIS) standardisiert (→ Als Basis Version 3.1 der MQTT-Spezifikation → 2018 Version 5 → komfortabler für Entwickler)
- Die **MQTT-Spezifikation unterscheidet TCP/IP-basierte und Nicht-TCP/IP-Netzwerke**.
  - **MQTT-SN** ist ausgelegt für eingebettete Geräte in non-TCP/IP-Netzwerken, wie zum Beispiel ZigBee, und ist optimiert für die Nutzung von Sensor- und Aktor-Lösungen (z.B. WSN). (→ Version 1.2 für Sensorgeräte)
- Eine der **Besonderheiten** von MQTT ist **die bandbreitenschonende Push-Kommunikation** aller Teilnehmer und die **Einfachheit in der Implementierung und Nutzung von Clients**.

# MQTT – Funktionsweise

- Die Architektur von MQTT basiert auf dem **Publish/Subscribe-Muster**.
- Die **Produzenten von Nachrichten (Publisher)** und die **Empfänger der Nachrichten (Subscriber)** sind mittels einem **MQTT Broker entkoppelt**.
- Der **MQTT Broker** ist verantwortlich, **eine Nachricht den richtigen Empfängern** sofort oder nach einer Neuverbindung des Clients **zuzustellen**, und er **verwaltet die Sessions** der verbundenen und nicht verbundenen Clients.
- Der **Broker** ist in der Lage, **eine einzelne, von einem Publisher versendete Nachricht**, an eine **Vielzahl von Subscribern gleichzeitig** zu versenden.
- Sowohl sendende als auch **empfangende Clients bleiben ständig mittels einer stehenden TCP-Verbindung mit dem MQTT Broker verbunden**, weshalb der MQTT Broker eine **Nachricht im Push-Verfahren ohne Verzögerung an Clients** ausliefern kann.
- **Mehrere Geräte können eine Subscription am MQTT Broker** erstellen und signalisieren über sogenannte **Topics, an welchen Nachrichten sie jeweils interessiert** sind.
- Der **Broker verwaltet die Subscriptions der einzelnen Clients** und ist in der Lage, neu ankommende Nachrichten direkt weiterzuleiten. Neben **Open Source MQTT Brokern wie mosquitto** gibt es hochskalierbare und erweiterbare Broker für den **professionellen Projekteinsatz wie HiveMQ**.

# MQTT – Funktionsweise



## Client

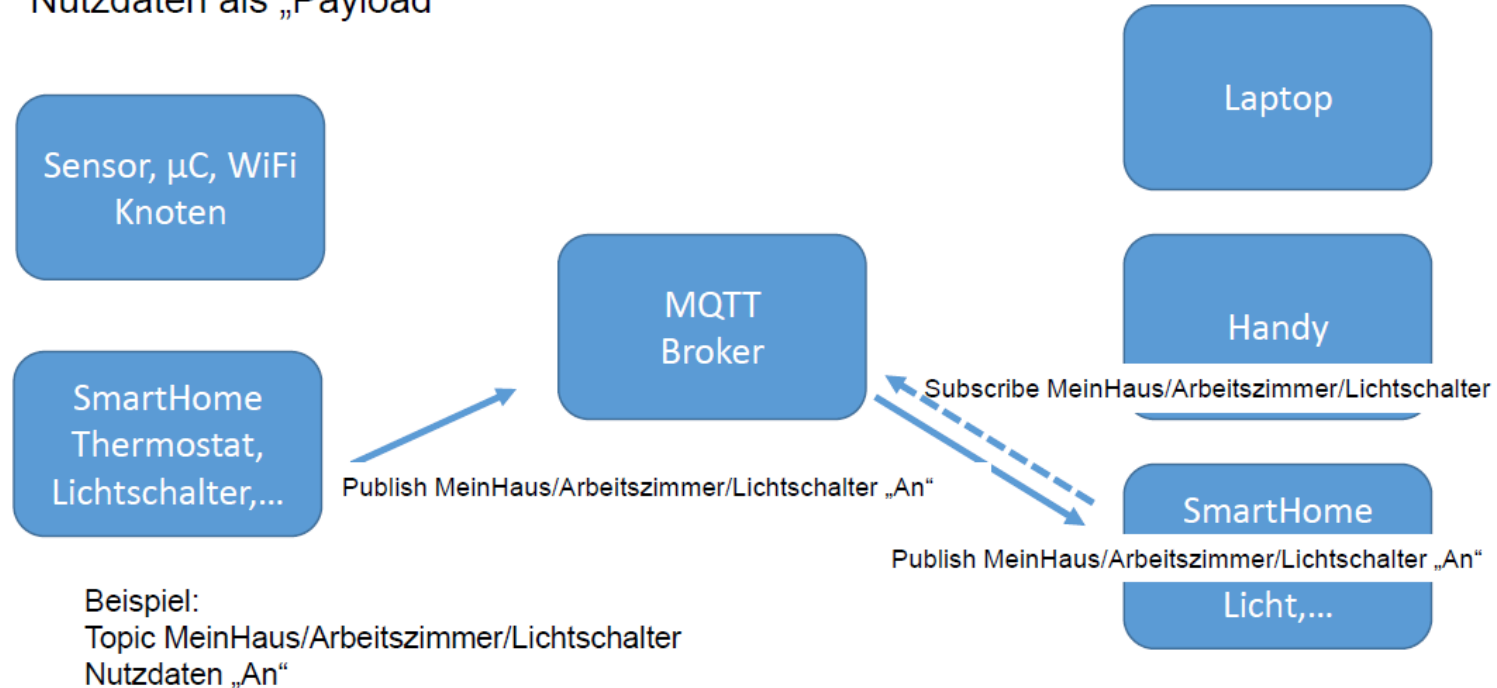
Beim Publish-Subscribe-Pattern sind zwei Arten von Clients vorhanden. Der Client, der Daten an den Broker sendet, nennt sich *Publisher (Producer)*. Ein Client, der sich beim Broker subskribiert und von diesem Nachrichten erhalten will, heißt *Subscriber (Consumer)*.



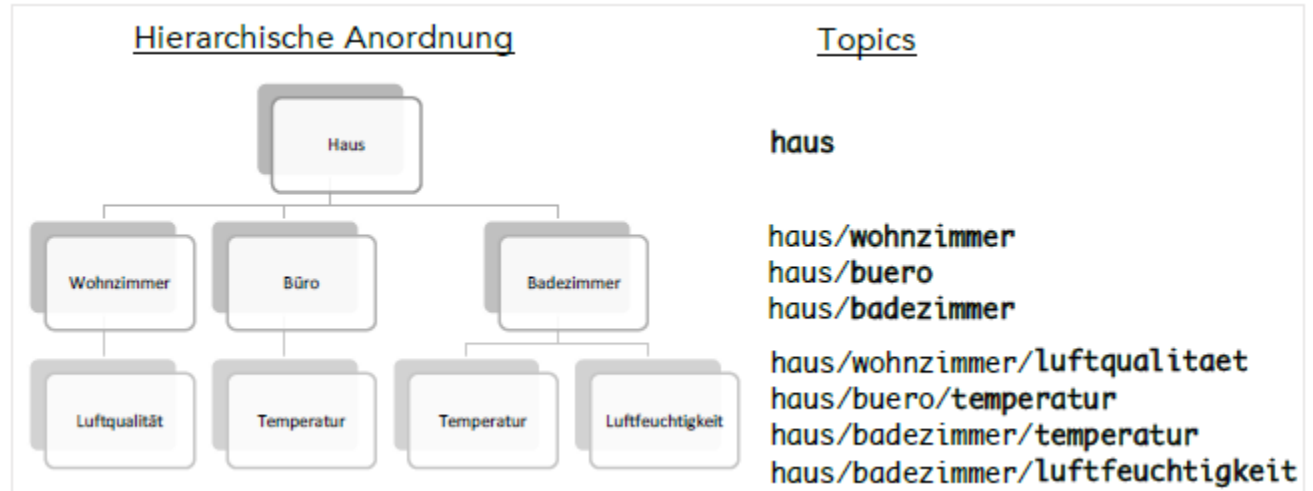
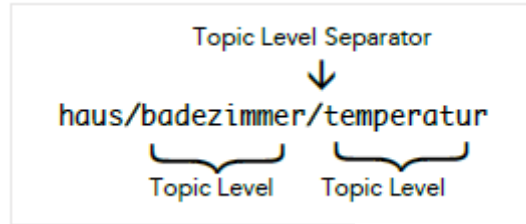
# MQTT – Funktionsweise: Beispiel Lichtschalter

Adressierung über Topic („Betreff“)

Nutzdaten als „Payload“



# MQTT – Adressierung durch Topics



# MQTT – Adressierung mit Filterung

- **Wildcards** ermöglichen es Subscribern, durch das Erstellen von Topic Filtern nur Topics zu abonnieren, die auf den Filter passen.

Single-Level Wildcard



haus/+/helligkeit

→ haus/wohnzimmer/helligkeit  
→ haus/buero/helligkeit  
→ haus/badezimmer/helligkeit  
haus/badezimmer/luftfeuchtigkeit

Multi-Level Wildcard



haus/badezimmer/#

haus/wohnzimmer/helligkeit  
haus/buero/helligkeit  
→ haus/badezimmer/helligkeit  
→ haus/badezimmer/luftfeuchtigkeit

Serverspezifische Statusinformationen:

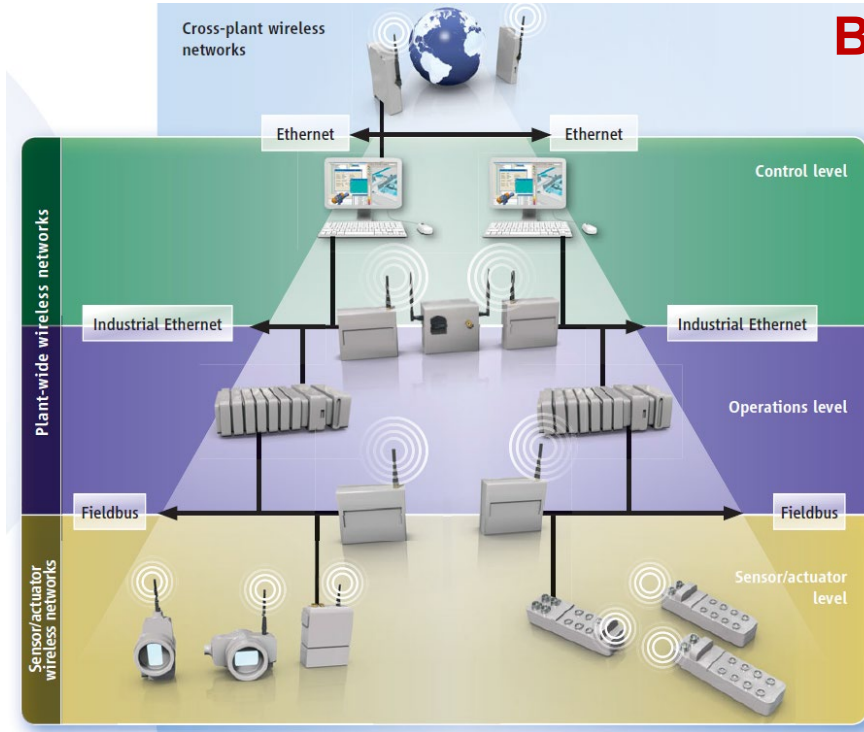
\$SYS/broker/clients/connected  
\$SYS/broker/messages/received  
\$SYS/broker/uptime  
\$SYS/broker/version

# MQTT – Nachrichtenarten: 14 Kontrollpakete

Name	Wert	Richtung	Beschreibung
Reserviert	0	Verboten	Reserviert
CONNECT	1	Client zum Server	Client Request, um mit Server zu verbinden
CONNACK	2	Server zum Client	Connect Bestätigung
PUBLISH	3	Client zum Server oder Server zum Client	Publish Nachricht
PUBACK	4	Client zum Server oder Server zum Client	Publish Bestätigung
PUBREC	5	Client zum Server oder Server zum Client	Publish empfangen (QoS 2, Teil 1)
PUBREL	6	Client zum Server oder Server zum Client	Publish freigegeben (QoS 2, Teil 2)
PUBCOMP	7	Client zum Server oder Server zum Client	Publish vollständig (QoS 2, Teil 3)
SUBSCRIBE	8	Client zum Server	Subscribe Anfrage
SUBACK	9	Server zum Client	Subscribe Bestätigung
UNSUBSCRIBE	10	Client zum Server	Unsubscribe Anfrage
UNSUBACK	11	Server zum Client	Unsubscribe Bestätigung
PINGREQ	12	Client zum Server	PING Anfrage
PINGRESP	13	Server zum Client	PING Antwort
DISCONNECT	14	Client zum Server	Client beendet Verbindung
Reserviert	15	Verboten	Reserviert

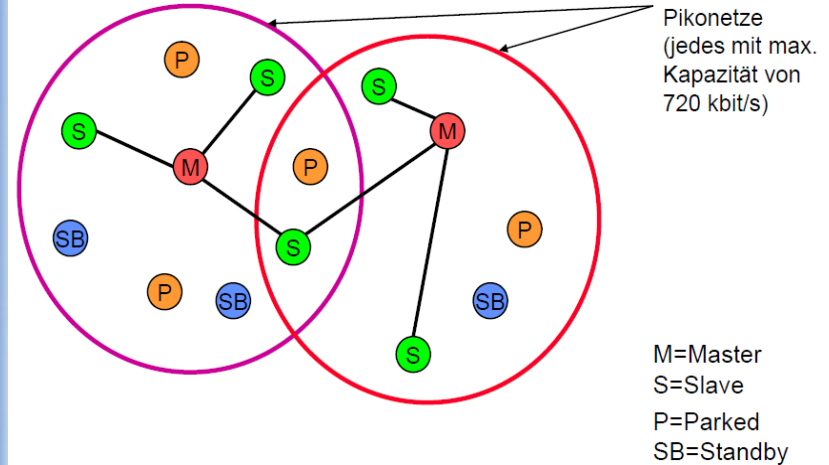
# Ausblick

## Industrial WLAN



## Bluetooth

## Zigbee



# UbiComp – Teil 8: Ethernet, OPC UA & MQTT

## Fragen?

Prof. Dr.-Ing. Dorothea Schwung